Unix Process Management

1. Unix Process:

- A process is an instance of running a program. A process is created or started whenever the user gives a command. When we tried out the ls command to list directory contents, a new process will start.
- Unix identifies every process by a Process Identification Number (pid) which is assigned when the process is initiated. When we want to perform an operation on a process, we usually refer to it by its pid.
- When we start a process there are two ways to run it.

  1.Foreground process

  2.Background process

2. Foreground Process:

- Every process that we start running in the unix is by default foreground process only. In foreground process it gets its input from keyboard and sends its output to screen.
- For example, if we are running the ls command, a new process will start as foreground process.
- While a program is running in foreground and taking much time, we cannot run any other commands because prompt would not be available until program finishes its processing and comes out.

3. Background Process:

- The process which is running at background is called a background process.
- To make a process as background process, the simplest way is to add an ampersand (&) at the end of the command.
- For example to list all the files with an extension of .doc and runs that process in background, you can mention the command as follows.

  $ls *.doc &
- The above ls command wants any input , it goes into a stop state until move it into the foreground and give it the data from the keyboard.
- The advantage of running a process in the background is that you can run other commands and you do not have to wait until it completes to start another.

- You can suspend the background process with kill command.

  $kill %job number Or $kill -9 %job number

4. Make background process as foreground process:

- The process is running in the background and it requires any input from keyboard then we can make that process as foreground process with following command.

  $fg %job number

- Every process has a job number and process id(pid), To make process from background to foreground or vice versa.

  Ex: $sleep 50 &

  [1] 927

  $fg %1

5. Make foreground process as background process:

- Every process is starting as foreground process until you make it as background.
- In order to make a process from foreground to background, you need to follow below two steps.

  1. Suspend the foreground process by enter CTRL+Z command.

  2. To run a process as background by run $bg %job number command.

  Ex: $ sleep 100

  CTRL+Z(for suspend process)

  $bg %1

- In the above example process start in a foreground then make it has background process.

6. Listing running processes:

- To list the current running processes in system, you can run the following commands.

  1) PS(Process Status):

  - The ps command is going to display the current running processes.

    Syntax : $ps [-option(s)]

  - The following options can be used along with ps command.

    $ps -f : shows more information about processes.

    $ps -a : shows information about other users' processes and own.

.

$ps -x : shows information about processes without terminals.

$ps u : shows more information about all users.

2) Jobs:

- The jobs command is going to display the information about current running processes.

  Syntax: $jobs [-option(s)]

- The following options can be used along with jobs command.

  $jobs -l : lists process IDs in addition to the normal information.

  $jobs -p : lists only process IDs of a processes.

  $jobs -r : lists only the running processes.

7.Terminating a process:

- We have two different ways to terminate a process based on whether it is a foreground process or background process.
- If the process is in foreground, we can terminate it by using ctrl+c command.
- If a process is in background, first we have to get its job number/id using ps command.

  $ps -f
- The above ps command gives the job number/id of a process.After that we can terminate the process with kill command.

  $kill job id/number

Note: If a process ignores a regular key command then we can use the following command.

$kill -9 job id/number

CUT COMMAND

Cut command is used for text processing, to select a portion of text by specifying the range or characters.

- Assume a file named as list.txt with the following content.

  $ cat list.txt

  cat for display file content

  cp for copy the files

  mkdir for creating a directory.

- We can displays 2nd character from each line of a file list.txt like the following.

  $ cut -c2 list.txt

  a

  p

  k

  To print the characters in a line, use the -c option in cut command.

- Select more than one character at a time from each line of a file.

  $ cut -c2,5 list.txt

  af

  po

  Kr

  The above command prints the second and fifth character in each line.

- Print a range of characters in a line by specifying the start and end position of the characters.

  $ cut -c2-8 list.txt

  at for

  p for c

  kdir fo

  The above command print characters from second position to eighth position from each line of a file.

- To print characters from specified start position to end of each line from list.txt file.

  $ cut -c3- list.txt

  t for display file content

  for copy the files

  dir for creating a directory.

- To print characters from start to specified end position of each line from list.txt file.

  $ cut -c-8 list.txt

  cat for

  cp for c

  mkdir fo

- Select a specified field from each line of a file.

  $ cut -d" " -f3 list.txt

  display

  copy

  Creating

  - '-d' option is used to specify the delimiter.
  - '-f' option is used to specify the field position.

  The above command prints the third field in each line by treating the space as delimiter.

- Select multiple fields from file, print the first and third field from each line of a file.

  $ cut -d" " -f1,3 list.txt

  cat display

  cp copy

  mkdir creating

- Select range of fields from each line of a file.

  $ cut -d" " -f2-4 list.txt

  for display file

  for copy the

  for creating a

- Select fields from specified start position to end of each line from list.txt file.

  $ cut -d" " -f4- list.txt

  file content

  the files

  a directory.

- Select fields from start position to specified end position of each line from list.txt file.

  $ cut -d" " -f-4 list.txt

  cat for display file

  cp for copy the

  mkdir for creating a

FIND COMMAND

The find command is used to search and locate list of files and directories based on conditions you specify for files that match the arguments.

1.To list all files in current and sub directories

  $find .

  Or

  $find . -print

2.Search specific directory or path

  $find ./unix

The above command list all the files of unix directory and sub directories.

3.To find all the files named with sample.txt in current directory.

```
$find  .  -name  sample.txt
```

In the above command . represents current directory and -name option is used to find the file.

4.To find all the files named with sample.txt in anywhere on the system.

```
$find  / -name  sample.txt
```

In the above '/' represents entire system.

5.To find for files in a specific directory.

```
$find ./unix  -name  sample.txt
```

The above command listing the files which named as sample.txt in a unix directory.

6.To list all files with .html extension of a current directory

```
$find  .  -name  "*.html"
```

7.To find files using name and ignoring case.

```
$find  .  -iname  sample.txt
```

The above command listing all the files whose name is sample.txt and contains both capital and small letters in current directory.

8.To find directories using directory name

```
$find  unix  -type  d  -name  Demo
```

The above command listing the directories whose name is Demo in unix directory.

9.To find html files using name

```
$find  .  -type  f  -name  sample.html
```

The above command listing all html files whose name is sample.html in a current working directory.

10.To find files with 777 permissions

```
$find  .  -type  f  -perm  777  -print
```

The above command listing all files whose permissions are 777 in a current directory.

11.To find files without 777 permissions

```
$find . -type f ! -perm 777
```

The above command listing all files whose permissions are not 777 in a current directory.

12.To find and remove single file

```
$find . -type f -name “sample.txt” -exec rm -f {} \;
```

13.To find and remove multiple files

```
$find . -type f -name “ *.txt ” -exec rm -f {} \;
```

14.To find all empty files

```
$find . -type f -empty
```

15.To find all empty directories

```
$find . -type d -empty
```

16.To find all hidden files

```
$find . -type f -name “.*”
```

17.To find all the files which are modified last 50 days back.

```
$find . -mtime 50
```

18.To find all the files which are modified last 50 to 100 days back.

```
$find . -mtime +50 -mtime -100
```

19.To find all the files which are accessed last 50 days back.

```
$find . -atime 50
```

20.To find all the files which are changed in last 1 hour.

```
$find . -cmin -60
```

21.To find all the files which are modified in last 1 hour.

```
$find . -mmin -60
```

22.To find all the files which are accessed in last 1 hour.

```
$find . -amin -60
```

23.To list all 50MB files in current directory.

```
$find . -size 50M
```

24.To find all the files which are greater than 50MB and less than 100MB.

```
$find . -size +50M -size -100M
```

25.To find single file based on user

```
$find / -user root -name sample.txt
```

The above command find a file called sample.txt under / root directory of owner root.